REMARKS

The Examiner found that claims 41, 43, and 45 would be allowed if written in independent form to include the requirements of the base and intervening claims. Applicants amended these claims to include the requirements of the base claims (there are no intervening claims) and submit that these claims are now in condition for allowance.

Applicants amended claims 1, 10, and 19 to include the requirements of claims 40, 42, and 44 and cancelled claims 40, 42, and 44.

The Examiner rejected canceled claims 40, 42, and 44 depending from claims 1, 10, and 19, whose requirements are now in independent claims 1, 10, and 19, as obvious (35 U.S.C. §103) over Harper (U.S. Patent No. 5,765,165) and Bereznyi (U.S. Patent No. 6,449,695). Applicants traverse for the following reasons.

Amended independent claims 1, 10, and 19 concern processing an input file in a file system, wherein the input file has an input file name, by: storing in cache memory a data structure generated by applying a function to all file names in a file system to determine values corresponding to the file names, wherein the data structure indicates those values corresponding to the file names to indicate all file names used in the file system; applying a function to map the input file name to a value; and scanning the data structure in cache memory without reading directory information from storage locations in a storage device to determine whether there is a preexisting file in the file system having a name that maps, according to the function, to the same value to which the input file name maps, wherein two files that map to a same value according to the function are capable of having a same name.

The Examiner recognized that Harper fails to tech storing the data structure indicating all file names in the file system in cache memory, wherein the data structure is scanned in cache memory to determine whether there is a preexisting file without reading directory information from storage locations in the storage device, but instead cited Bereznyi, col. 26, lines 34-47 as teaching the storage of a hash table in cache memory. The Examiner found that it would be obvious to modify the hash of Harper to store in cache based on Bereznyi. (Office Action, pg. 3). Applicants traverse.

The cited Bereznyi mentions a hash index used to access and identify data items within a cache. Thus, the hash index of Bereznyi is entirely different than the claimed data structure that indicates those values corresponding to the file names to indicate all file names used in the file system. Further, nowhere in the cited combination is there any suggestion of storing in cache memory a data structure to allow a determination of whether there is a preexisting file in a file system without reading directory information from storage locations in a storage device as claimed. Applicants submit that the Examiner is using hindsight to modify Harper with Bereznyi to allow the caching of the hash table of Harper for the claimed purposes.

Further, even if one were to modify Harper to store the Harper hash table in cache, the cited Bereznyi does not overcome the deficiencies of Harper in teaching or suggesting the claimed data structure that is generated by applying a function to all file names in a file system. Instead, the cited Harper (col. 8, lines 20-35, col. 12, lines 17-29) on page 3 of the Office Action discusses applying a hash to linked lists of inodes to check for duplicates in a linked list of inodes. An inode comprises a data structure providing information on a file and pointer to the physical block having the file data. Applicants submit that an inode is different from a file name.

Moreover, Harper teaches away from a data structure to indicate all file names in the file system, because Harper notes that an inode has all information about a file except the file name. (Harper, col. 1, lines 20-23). Thus, Harper expressly teaches away from the claim requirement of a data structure generated by applying a function to all file names in a file system to determine values corresponding to the file names because the hash in Harper is used to determine duplicate inodes, which expressly does not include information about file names as claimed.

Moreover, because Harper expressly teaches away from file names, Harper also teaches away from the claim requirement of scanning the data structure in cache memory without reading directory information from storage locations to determine whether an input file name maps to a preexisting file name in the file system. Thus, combining Harper with Bereznyi does not teach the claimed combination.

The Examiner cites to col. 2, line 57 to col. 3, line 16 of Harper to assert that Harper teaches that the cited inode hash can apply to determine duplicate file names. (Office Action,

pgs. 3-4) Applicants traverse. The cited cols. 2-3 discusses using hashing to facilitate storage and retrieval of data, and that a plurality of identifiers are partitioned into buckets having slots for storage of data, where the buckets available for storage are identified by a set of bucket identifiers. According to Harper, the bucket identifiers may be mapped by hashing to values.

Although the cited cols 2 and 3 of Harper discuss using hashing to facilitate data storage, nowhere does the cited Harper anywhere teach or suggest the claim requirement of generating a data structure by applying a function to all file names in a file system to determine values corresponding to all file names, where the data structure indicates the values to indicate all the file names used in the file system.

To the extent the Examiner is trying to modify the cited Harper to teach this claim requirement, such proposed modifications are improper because there is no teaching or suggestion anywhere in the cited Harper that a function, such as a hash, be applied to all file names to produce values in a data structure to indicate all file names used in the file system. Instead, the cited Harper discusses applying hashes to inodes and bucket identifiers, not all file names in a file system to produce a data structure that indicates all the names used in the file system.

Applicants submit that the Examiner is relying on inappropriate use of hindsight to modify Harper to produce the claimed data structure generated by applying a function to all file names in a file system. The Manual of Patent Examination Procedure (MPEP) states that the "mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination." MPEP 2143.01, pg. 2100-136 (8th Ed., Rev. 1, Feb. 2003). Applicants note that even modifications that may appear simple cannot be made unless the Examiner provides suggestion or motivation to make the apparent "simple" modification. According to the Federal Circuit,

> In a proper obviousness determination, "[w]hether the changes from the prior art are 'minor', . . . the changes must be evaluated in terms of the whole invention, including whether the prior art provides any teaching or suggestion to one of ordinary skill in the art to make the changes that would produce the patentee's . . . device." Northern Telecom, Inc. v. Datapoint Corp., 908 F.2d 931, 935, 15 USPQ2d 1321, 1324 (Fed. Cir.), cert. denied, 498 U.S. 920 (1990). This includes what could be characterized as simple

> changes, as in In re Gordon, 733 F.2d 900, 902, 221 USPQ 1125, 1127 (Fed. Cir. 1984) (Although a prior art device could have been turned upside down, that did not make the modification obvious unless the prior art fairly suggested the desirability of turning the device upside down.).

In re Chu, 36 USPQ2d 1089, 1094 (Fed. Cir. 1995) (emphasis added)

Here, the examiner has not cited any art that suggests modifying the hash of inodes of Harper to determine duplicates to all file names in a file system. Moreover, Harper teaches away from such a modification because Harper mentions that inodes contain information on a file except the file name. (Col. 1, lines 20-25) For these reasons, the cited prior art fails to teach or suggest the modification the Examiner is proposing to modify the inode hash of Harper to apply to all file names in a file system and to store the hash of Harper in cache to allow for scanning in cache memory without reading directory information from storage locations in a storage device to determine whether there is a preexisting file in the file system.

Moreover, the proposed modification of Harper would also be inappropriate because it would change the principle of operation of the cited Harper. The MPEP states that:

> If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims prima facie obvious.

MPEP, Sec. 2143.02, pg. 127.

Harper does its hash "by allocating a relatively short block of memory, such as one byte, and by hashing inode identifications to one of the bits of the block of memory." (Harper, col. 4, lines 1-5) The claimed data structure requires indication of hash values corresponding to all file names in a file system and would require far more than one byte of memory allocated because a file system as known in the art must accommodate thousands of file names. See, Application, pg. 6, line 24 to pg. 7, line 14.

Applicants submit that in this case, the proposed modification of Harper to apply to file systems would change the principle of operation by requiring the allocation of memory for the hash bit map far exceeding the one byte Harper discusses in the "Summary of the Invention" section and far exceeding the hash needed to represent elements in a linked list. (Harper, col. 3, line 65 to col. 4, line 5) Thus, Harper teaches away from applying the discussed hash technique

for checking duplicates in a linked list to duplicates of all files in a file system because Harper discusses that his "present invention" allocates a short block of memory, such as one byte, as the hash bitmap used to indicate identifiers elements in a linked list.

For the above reasons, Applicants submit that the amended claims 1, 10, and 19 distinguish over the art cited Harper and Bereznyi.

The Examiner rejected claims 2-9, 11-18, 20-36 as obvious over Harper. Applicants submit that these claims are patentable over the cited Harper because they depend from one of amended claims 1, 10, and 19, respectively, which are patentable over the cited combination for the reasons described above. Further claims discussed below provide additional grounds of patentability over the cited art.

Claims 4, 13, and 22 depend from claims 3, 12, and 13, which require that the data structure includes an entry for each possible integer value capable of being generated from the hash function. Claims 4, 13, and 22 further require that processing the data structure to determine whether there is a preexisting file comprises determining whether the entry for the integer value to which the input file name maps indicates the presence of one preexisting file mapping to the same integer value as the input file name.

The Examiner cited col. 8, lines 25-35 of Harper as teaching the additional requirements of these claims. (Office Action, pg. 8) Applicants traverse.

As discussed, the cited col. 8 discusses determining whether the identifier of an element to add to a linked list is a duplicate of an element in the linked list. Nowhere does the cited Harper anywhere teach or suggest processing the data structure to determine whether there is a preexisting file in the file system having the same name as the input file name based on the entry for the integer value as claimed. Instead, Harper concerns duplicates for elements on a linked list and nowhere teaches, suggests or remotely using a function and data structure as claimed to determine whether an input file name has a same name as a preexisting file in the file system.

For these reasons, claims 4, 13, and 22 provide additional grounds of patentability over the cited combined art.

Claims 6, 15, and 24 depend from claims 1, 10, and 19 and further require applying the function to each file name in the file system to map each file name to one value and indicating in the data structure, for each file name, that there is one preexisting file for the value to which the file name maps.

The Examiner cited col. 8, lines 20-28 of Harper as teaching the additional requirements of these claims. (Office Action, pg. 8) Applicants traverse.

The cited Harper only discusses applying a hash to elements on a linked list. Nowhere does the cited Harper anywhere teach or suggest applying a function, or the hash of Harper, to each file name in a file system as claimed to indicate whether, for each file name, there is one preexisting file. Instead, as discussed, the cited Harper only discusses indicating in the hash bit map identifiers elements in a linked list.

For these reasons, the cited references do not teach or suggest, alone or in combination, the requirements of claims 6, 15, and 24.

Claims 7, 16, and 25 depend from claims 6, 15, and 24 and further require that the input file is the subject of an access request. Further, each file in the file system is scanned to determine if there is at least one preexisting file having the same name as the input file name if there is one preexisting file in the file system having a name that maps, according to the function, to the same value to which the input file name maps.

The Examiner cited col. 8, lines 29-35 of Harper as teaching the additional requirements of these claims. (Office Action, pgs. 9) Applicants traverse.

The cited col. 8 discusses comparing the identifier to add to the identifiers of elements on a list once a potential duplicate is identified. Nowhere does the cited Harper anywhere teach, suggest or remotely mention that the input file is the subject of an access request.

Again, the Examiner is modifying prior art in a manner nowhere taught or suggested in the cited art. The proposed modifications are based solely on the claim requirements and not suggested in any cited art. Although different elements of the claims may be separately discussed in the cited art, such as a file system, hash functions, etc., nowhere does the cited

Harper anywhere suggest the combination of these requirements as claimed to determine whether there is a preexisting file for a requested file in a file system.

Accordingly, the cited references do not teach or suggest, alone or in combination, the requirements of claims 7, 16, and 25.

Claims 8, 9, 17, 18, 26, and 27 depend from claims 7, 16 or 14 and provide further details on file system access operations based on using the claimed technique to determine whether there is a preexisting file having the same name as the input file subject to the access request. Because the Examiner has cited the same art in rejecting these claims (Office Action, pgs. 9-10), the cited combination fails to disclose the additional requirements of these claims in combination with the base and intervening claims from which they depend for the reasons discussed above, and because there is no suggestion in the cited art to modify Harper to use the discussed hash function with file names in a file system as claimed.

Claims 28, 31, and 34 depend from claims 1, 10, and 19 and further require searching the file system for one file having the same name as the input file name if the data structure indicates that one preexisting file has a name that maps, according to the function, to the same value to which the input file maps. An operation is performed if the file system includes one file having the same name as the input file.

The Examiner cited col. 8, lines 25-35 of Harper as teaching the additional requirements of these claims. (Office Action, pgs. 10-11) Applicants traverse.

The cited col. 8 discusses comparing the identifier to add to the identifiers of elements on a list once a potential duplicate is identified. Nowhere does this cited col. 8 anywhere teach, suggest or remotely mention searching a file system for a preexisting file having the same name as the input file as claimed.

Accordingly, claims 28, 31, and 34 provide further grounds of distinction over the cited art.

Applicants further submit that claims 29, 30, 32, 33, 35, and 36 are also patentable over the cited art because they depend from claims 28, 31, and 34, which are patentable over the cited art for the reasons discussed above, and because the additional requirements concerning the file

system operations are nowhere taught or suggested in the cited Harper, which only mentions adding elements to a linked list, not performing file system related operations as claimed.

1. Claims 37-39 are Patentable Over the Cited Art

The Examiner rejected claims 37-39 as obvious over Harper in view of Williams (U.S. Patent No. 5,990,810). Applicants traverse for the following reasons.

Applicants submit that added claims 37-39 are patentable over the cited art because they depend from one of claims 1, 10, and 19, which are patentable over the cited art for the reasons described above, and because the additional dependent limitations provide further grounds of patentability over the cited art. The Examiner cited col. 11, lines 25-45 of Williams as teaching the additional requirements of these claims. (Office Action, pgs. 12-13) Applicants traverse for the following reasons.

The cited col. 11 of Williams discusses wide hash functions whose output values are wider, such that the probability of any two randomly chosen blocks having the same hashed value is negligible.

Although the cited Williams discusses wide hashes in general, nowhere does the cited Williams anywhere teach or suggest that a wide has is used to minimize the likelihood that two file names would be hashed to a same hash value. Nowhere does the cited Williams anywhere teach or suggest applying a wide hash to all the file names of a file system to avoid the likelihood that two file names would hash to a same value.

Accordingly, claims 37, 38, and 39 provide additional grounds of patentability over the cited art.

Conclusion

For all the above reasons, Applicant submits that the pending claims pending claims 1-39, 41, 43, and 45are patentable over the art of record. Applicants submit herewith the fees for a two-month extension of time. Nonetheless, should any additional fees be required, please charge Deposit Account No. 50-0585.

The attorney of record invites the Examiner to contact him at (310) 553-7977 if the Examiner believes such contact would advance the prosecution of the case.

Dated: April 5, 2004

By: _____

David W. Victor
Reg. No.: 39,867

Please direct all correspondences to:

David Victor
Konrad Raynes & Victor, LLP
315 South Beverly Drive, Ste. 210
Beverly Hills, CA 90212
Tel: 310-553-7977
Fax: 310-556-7984